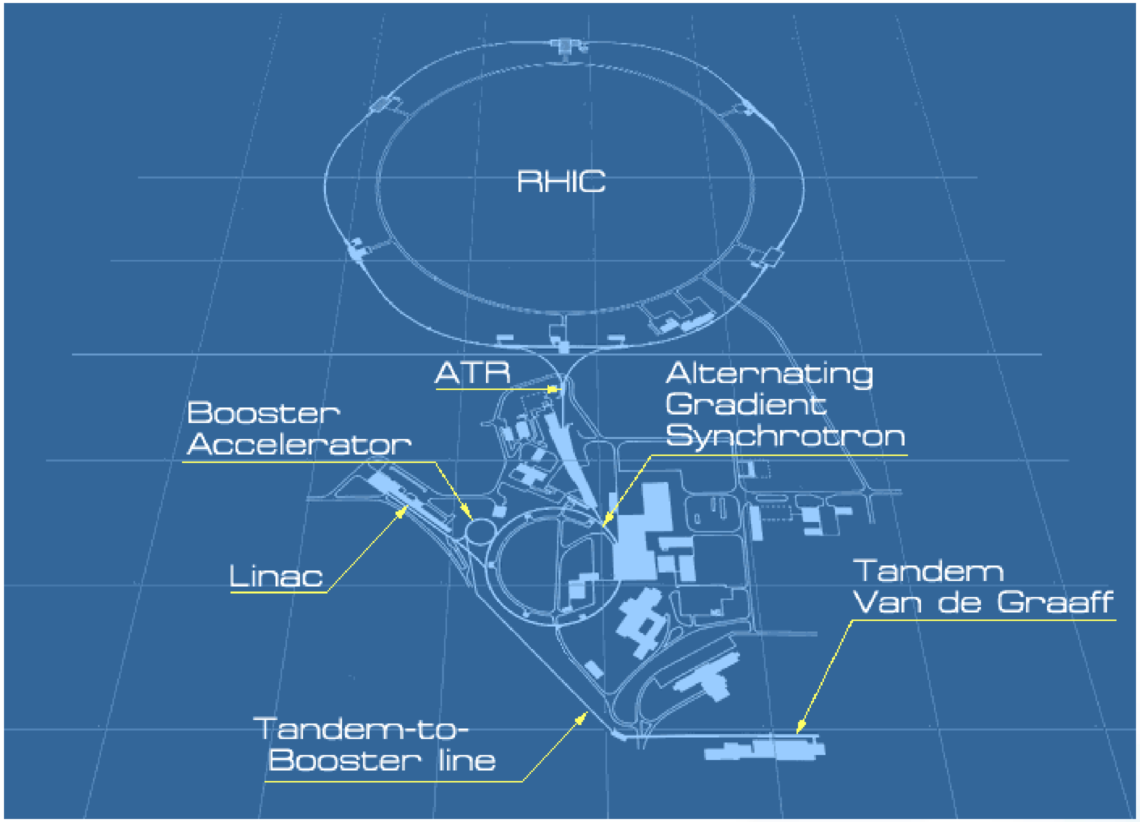
DSP Tutorial II

Real-life Implementation

BROOKHAVEN NATIONAL LABORATORY
a passion for discovery

U.S. DEPARTMENT OF ENERGY | Office of Science

Collider-Accelerator Complex



Kevin Smith
Tom Hayes
Freddy Severino
Kevin Mernick
Denny Nembhard
Geetha Narayan

Outline

- **High-level Algorithm flow**
- Simulink with XILINX System Generator
- Real life example – concept to proof
- Demo with HW-SW Co-Simulation
- Designs in development

Survey of System Designers

What do you use for high-level algorithm development?

- Mostly MATLAB
- MATLAB & Simulink
- Mostly C/C++
- None – just hand-code RTL or use IP


What do you use for high-level algorithm development?
Mostly MATLAB
38.2%
MATLAB & Simulink
29.1%
Mostly C/C++
12.7%
None – just hand-code RTL or use IP
20.0%

Objective

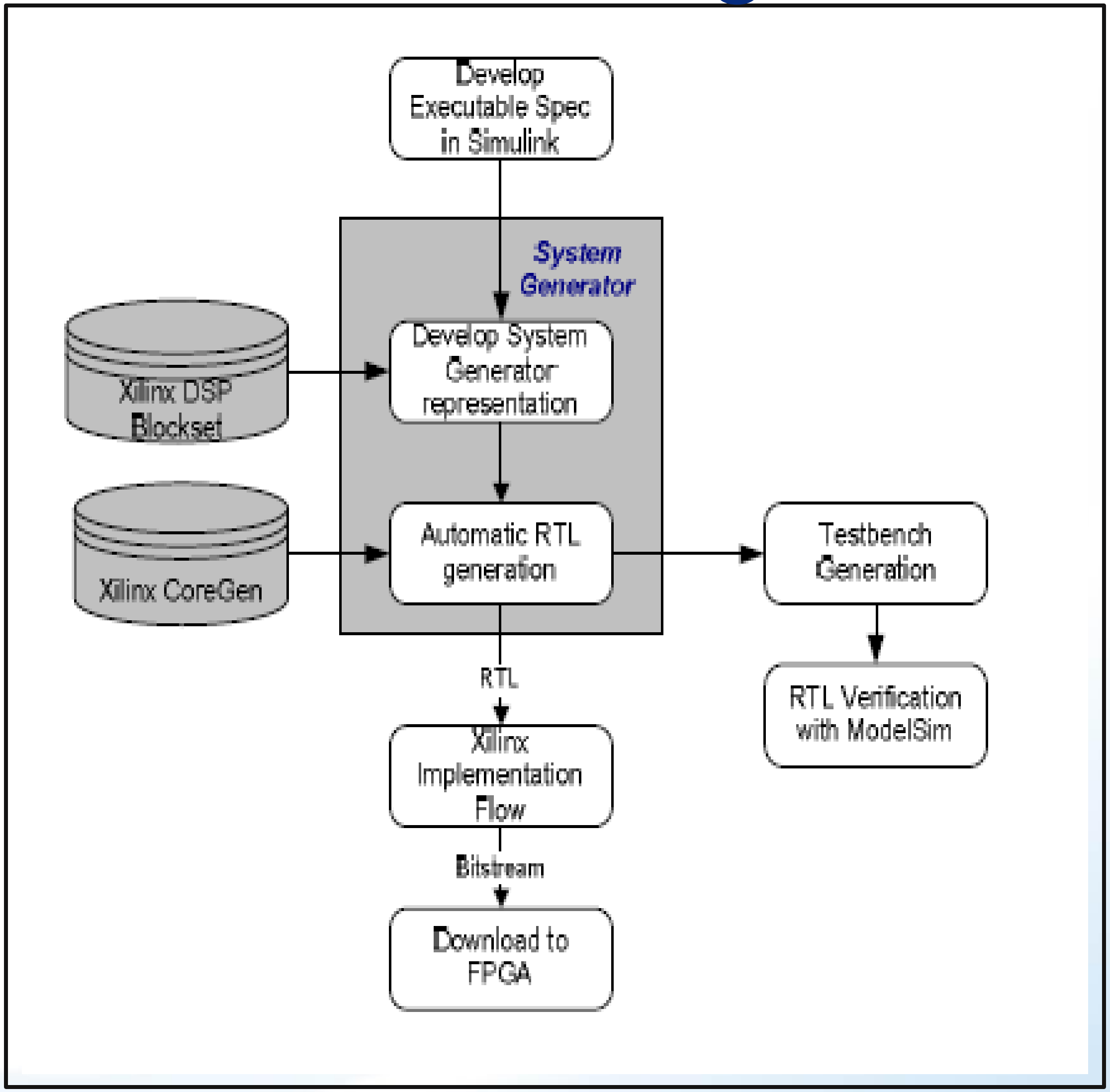
Everything is better when it's simple

- Demonstrate the integrated flow from system design to implementation of real-time DSP applications on FPGAs
    - Use MATLAB / Simulink to validate a simple DSP algorithm
    - Implement algorithm with XILINX System Generator
    - Verifying the DSP system using Simulink and HDL simulator
    - Preparing design for Co-Simulation on SP605 (Spartan-6) Board
    - Performing Hardware/Software Co-Simulation for the DSP system
- Developers with little FPGA design experience can quickly create FPGA implementations of DSP algorithms in a fraction of traditional RTL development times.

Outline

- High-level Algorithm flow
- **Simulink with XILINX System Generator**
- Real life example – concept to proof
- Demo with HW-SW Co-Simulation
- Designs in development

System Generator Design Flow



System Generator Features

- **DSP modeling**
  Xilinx blockset contains about 100 IPs with functions such as
  - signal processing (e.g., FIR filters, FFTs)
  - error correction (e.g., Viterbi decoder, Reed-Solomon encoder/decoder)
  - arithmetic, memories (e.g., FIFO, RAM, ROM), and digital logic
- **Bit and cycle accurate floating and fixed-point implementation**
- **Automatic code generation of VHDL or Verilog from Simulink**
  - Integrate legacy RTL
- **Hardware co-simulation**
  Validate working hardware and accelerate simulations using
  - Ethernet (10/100/Gigabit)
  - JTAG communication between a hardware platform and Simulink
- **Hardware / software co-design of embedded systems**
  Build and debug DSP co-processors for the Xilinx MicroBlaze™ soft processor core

Design Flow Strategies

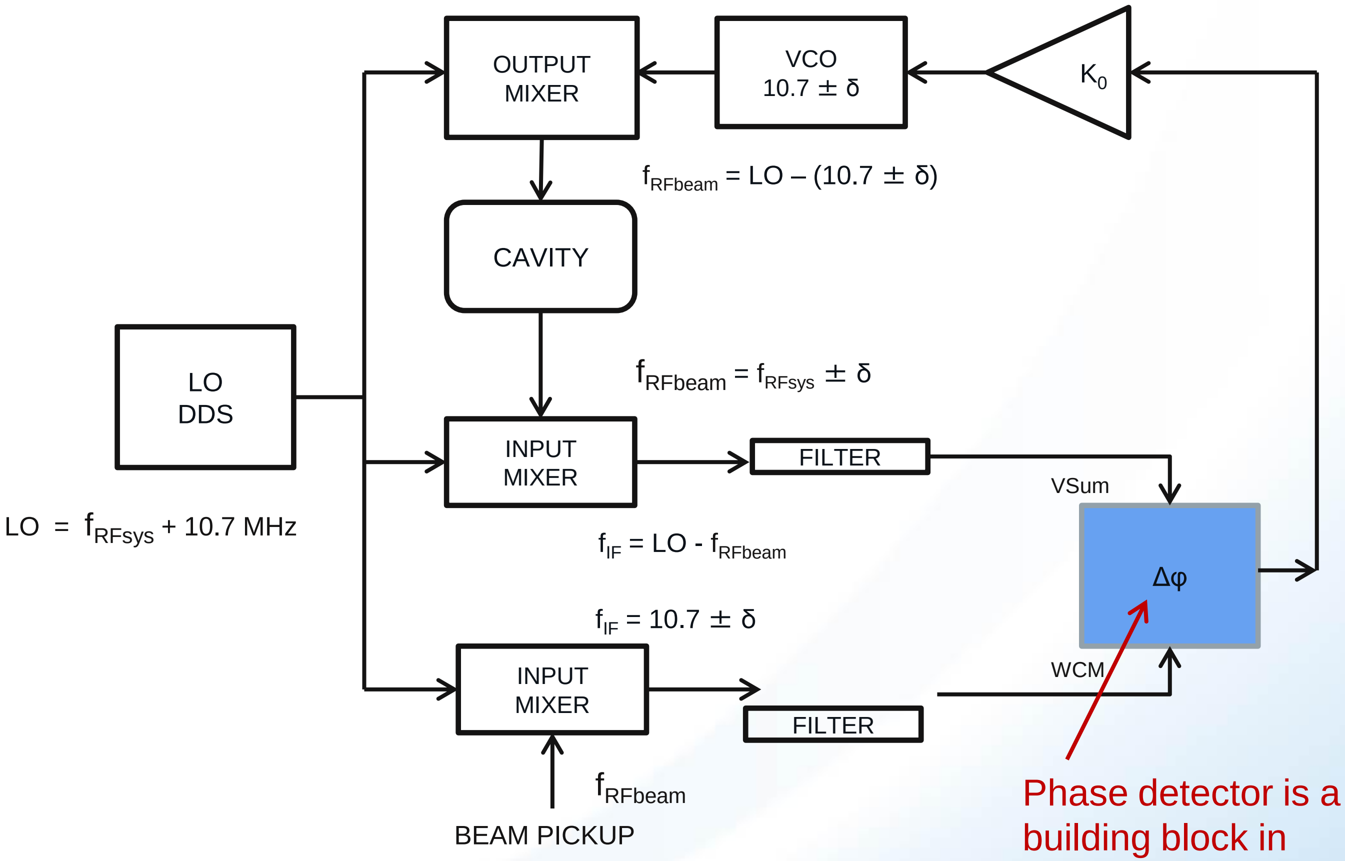
- **Algorithm Exploration**
  - useful for algorithm exploration and model analysis
  - for architecture exploration such as HW/SW partitioning
  - estimate the cost and performance of an implementation in hardware
- **Implementing Part of a Larger Design**
  - ideal to implement data paths and control
  - less suited for sophisticated external interfaces
  - design can be exported as an application specific IP to be integrated into a system
- **Implementing a Complete Design**
  - ‘Generate’ button instructs System Generator to translate the design
  - HDL that implements the design
  - HDL testbench that transforms results from Simulink simulations to be used in a logic simulator
  - scripts that guide downstream tools, such as XST for synthesis

Outline

- High-level Algorithm flow
- Simulink with XILINX System Generator
- **Real life example – concept to proof**
- Demo with HW-SW Co-Simulation
- Designs in development

Coherent Phase Detector for AGS Booster

Determine beam bunch phase with respect to net RF voltage per turn in a Heterodyne System

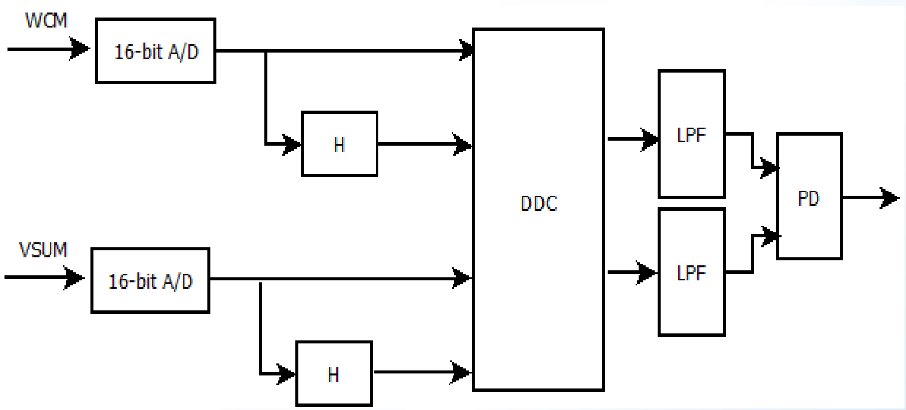


Phase detector is a main building block in loop control applications

Hilbert-transform Phase Detection Scheme

Digital phase detecting method making use of 90° phase shift property of Hilbert transform.

- Heterodyne signals are sampled by A/Ds
- An analytic signal is generated from a real signal by using the Hilbert transform
- An FIR filter is used for the approximation of the Hilbert transform
- Signals are down-converted to DC by mixing
- After LPF and decimation phase difference is computed



Mathematical Computation

Received signals :

$s_1(t) = a_1(t) \cos(\omega t + \Phi_1)$

$s_2(t) = a_2(t) \cos(\omega t + \Phi_2)$

Hilbert Transform is equivalent to a 90º phase shift linear filter

$H[s_1(t)] = a_1(t) \sin(\omega t + \Phi_1)$

$H[s_2(t)] = a_2(t) \sin(\omega t + \Phi_2)$

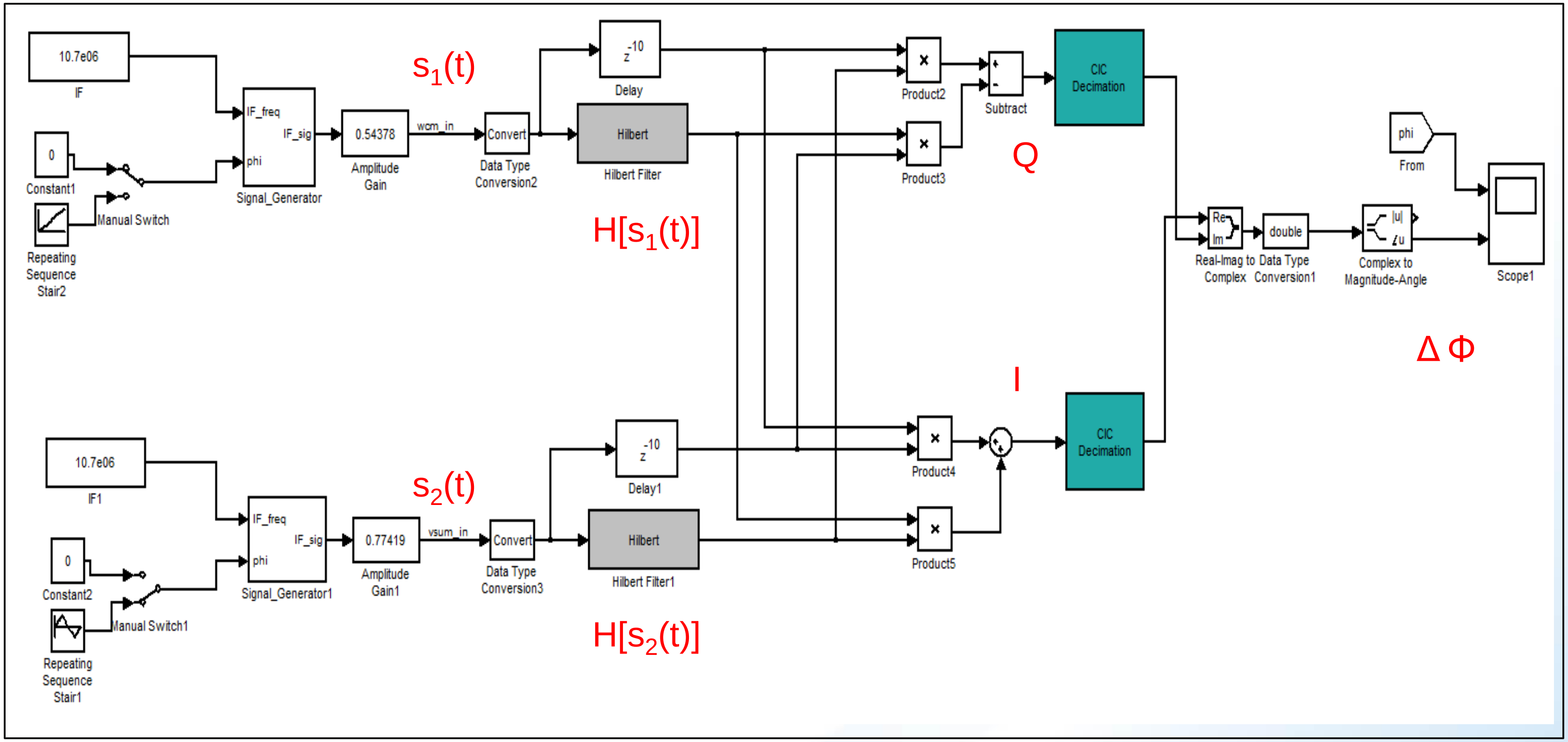
$$Q = s_1(t) \,.\, H[s_2(t)] - H[s_1(t)] \,.\, s_2(t)$$
$$= a_1(t) \,.\, a_2(t) \,.\, \sin(\Phi_2 - \Phi_1)$$

$$I = s_1(t) \,.\, s_2(t) + H[s_1(t)] \,.\, H[s_2(t)]$$
$$= a_1(t) \,.\, a_2(t) \,.\, \cos(\Phi_2 - \Phi_1)$$

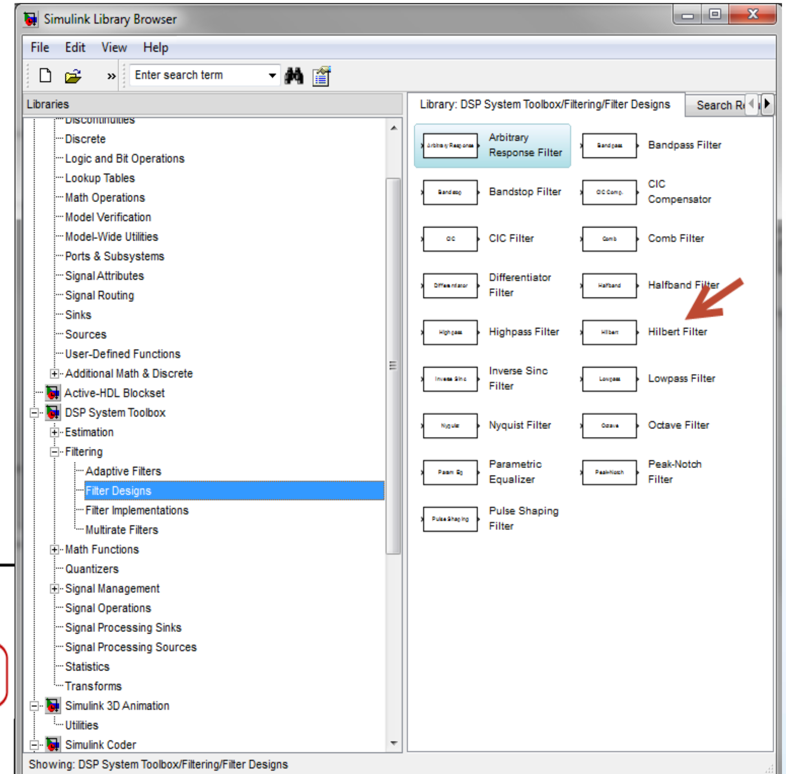
Phase Difference :

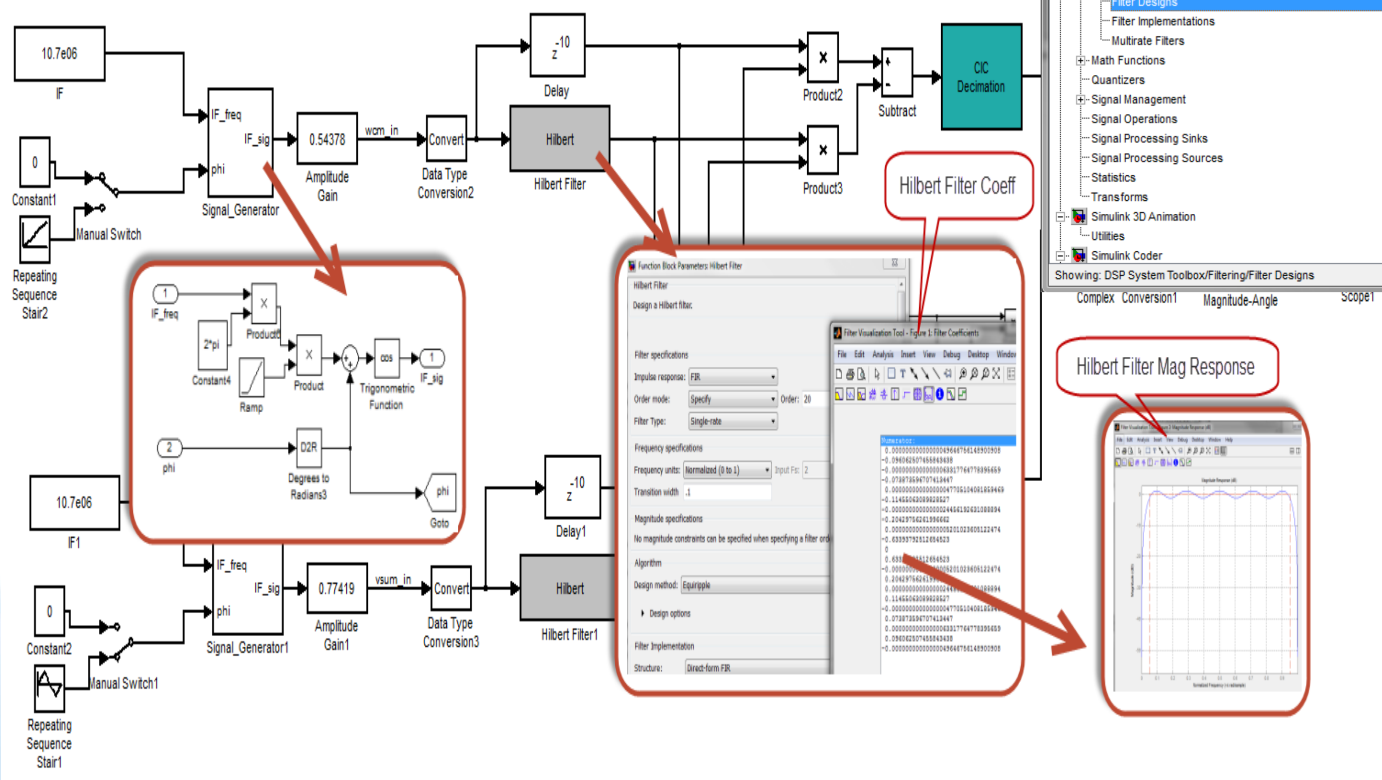
$$\Delta \Phi = (\Phi_2 - \Phi_1) = \tan^{-1}(Q / I)$$

AGS Phase Detector Simulink Model



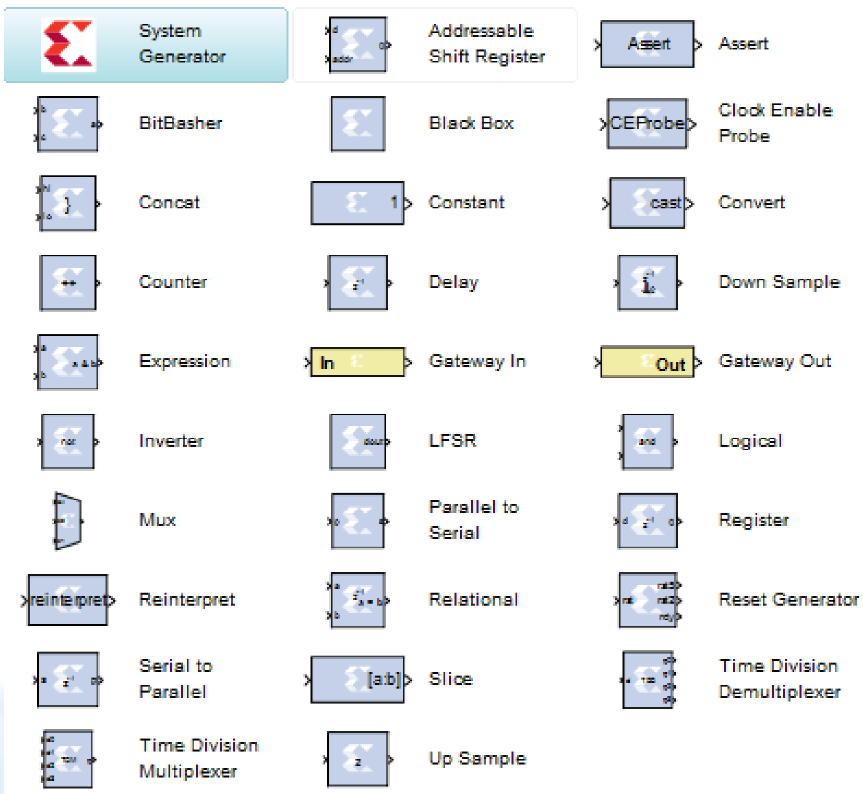
Simulink Toolbox & Test Bench



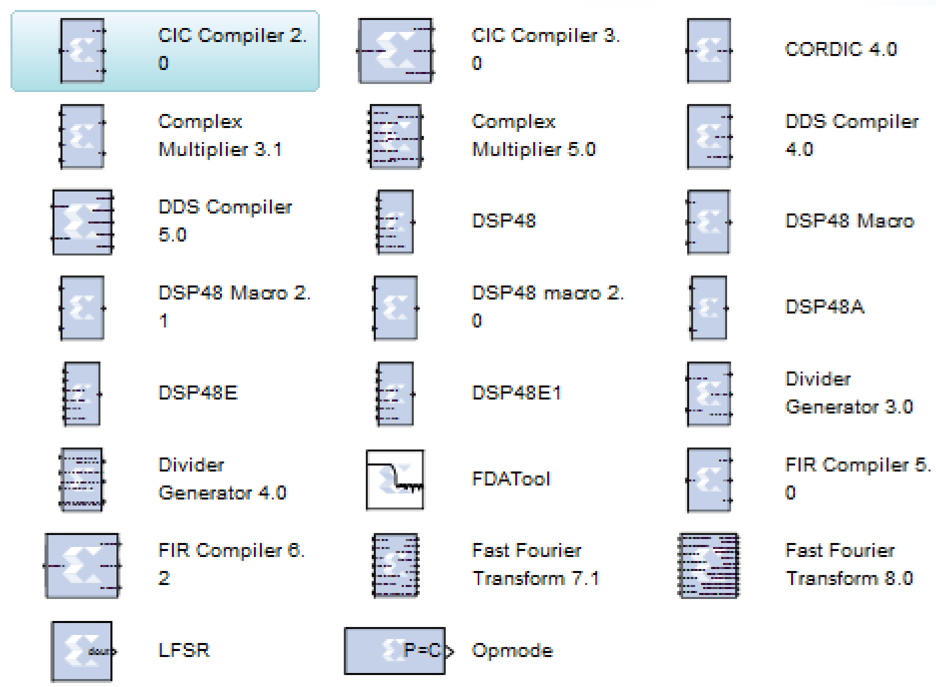


Xilinx library of bit and cycle-true models

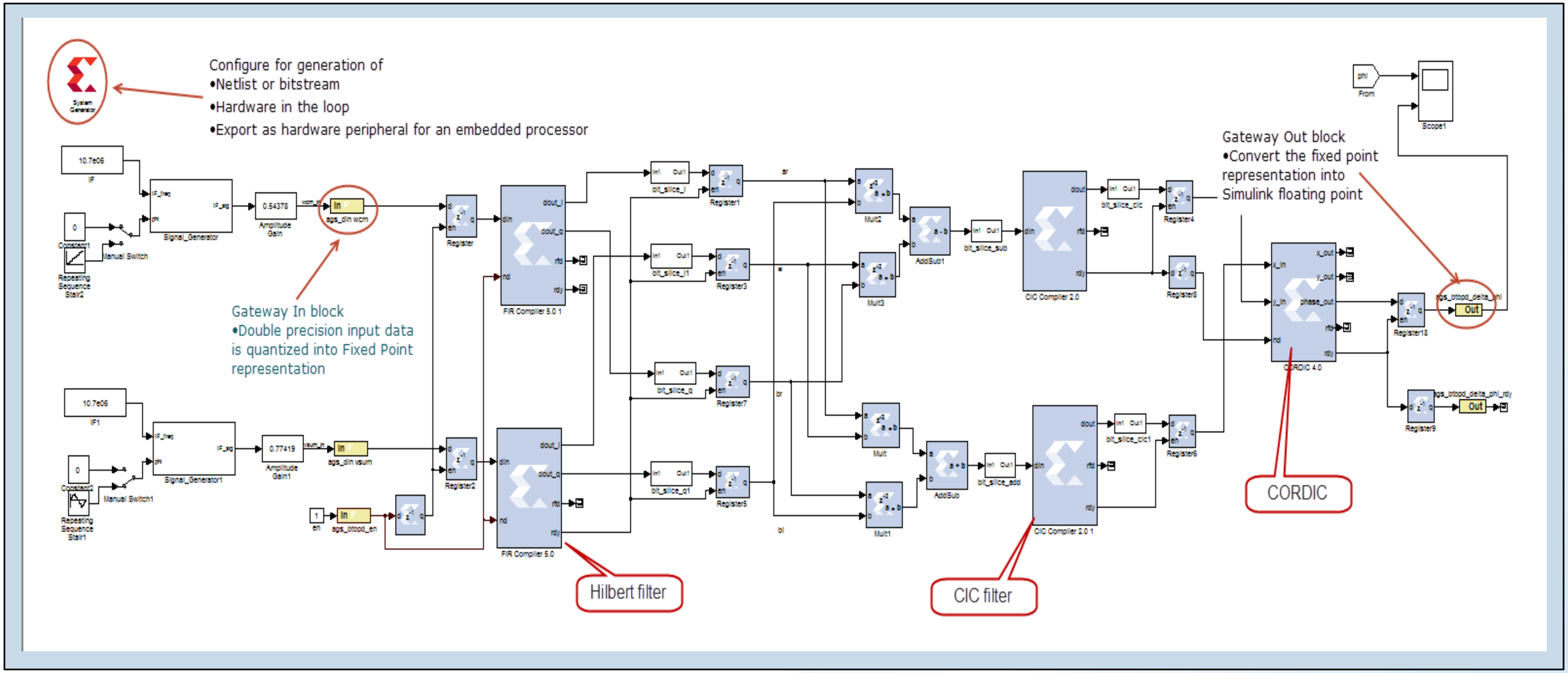
Basic Elements



DSP Blocks



AGS PD System Generator Model



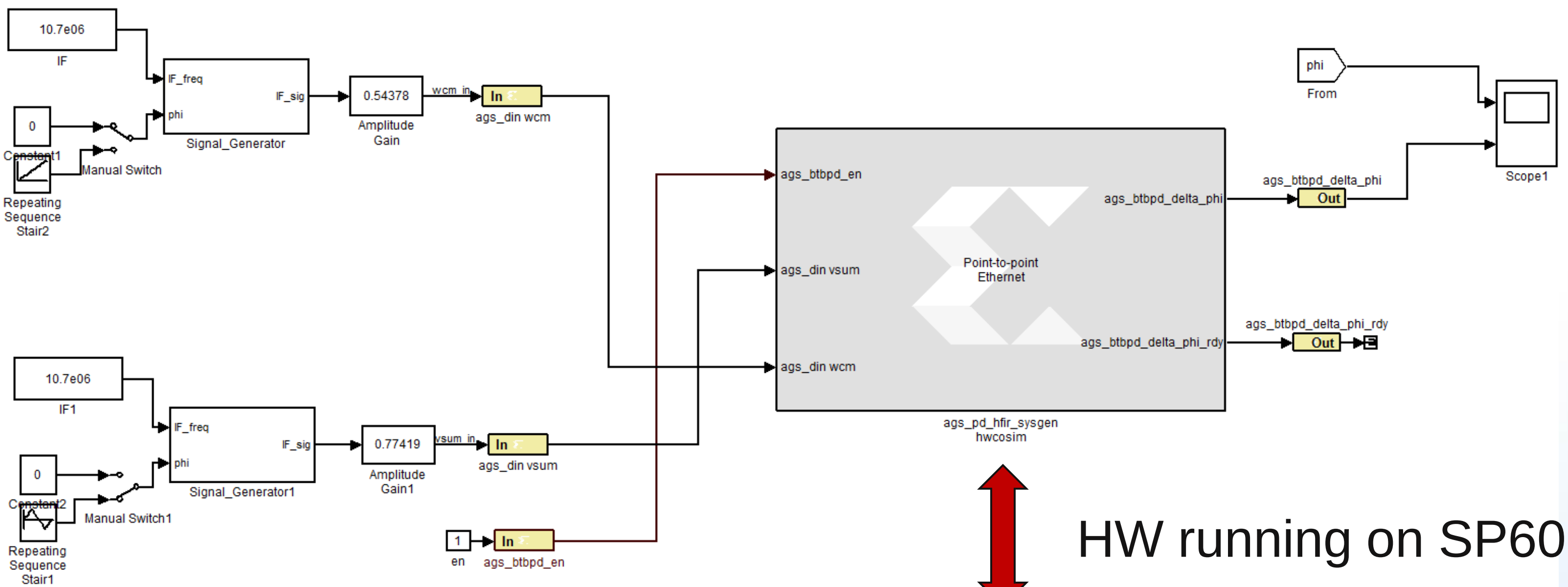
Video on Firmware Generation

(2 min)

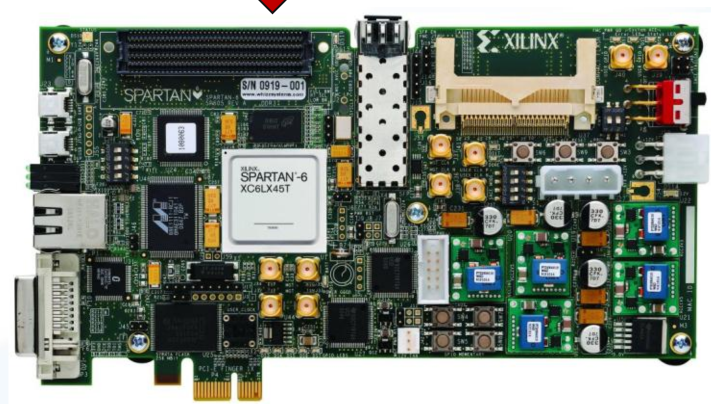
Outline

- High-level Algorithm flow
- Simulink with XILINX System Generator
- Real life example – concept to proof
- **Demo with HW-SW Co-Simulation (HW in the Loop)**
- Designs in development

HW / SW Co-Simulation



HW running on SP605

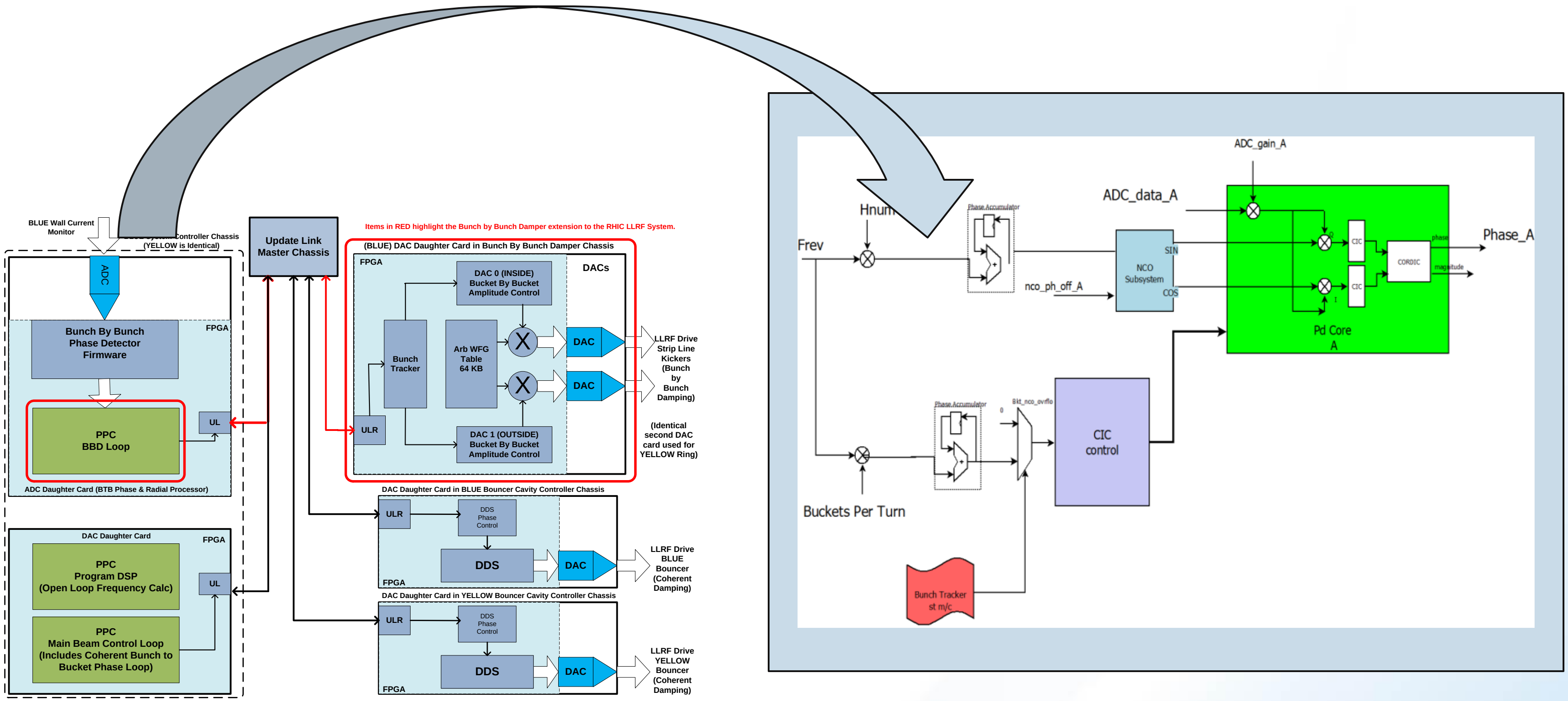


HW / SW interfaces handled automatically by System Generator

Outline

- High-level Algorithm flow
- Simulink with XILINX System Generator
- Real life example – concept to proof
- Demo with HW-SW Co-Simulation
- **Designs in development**

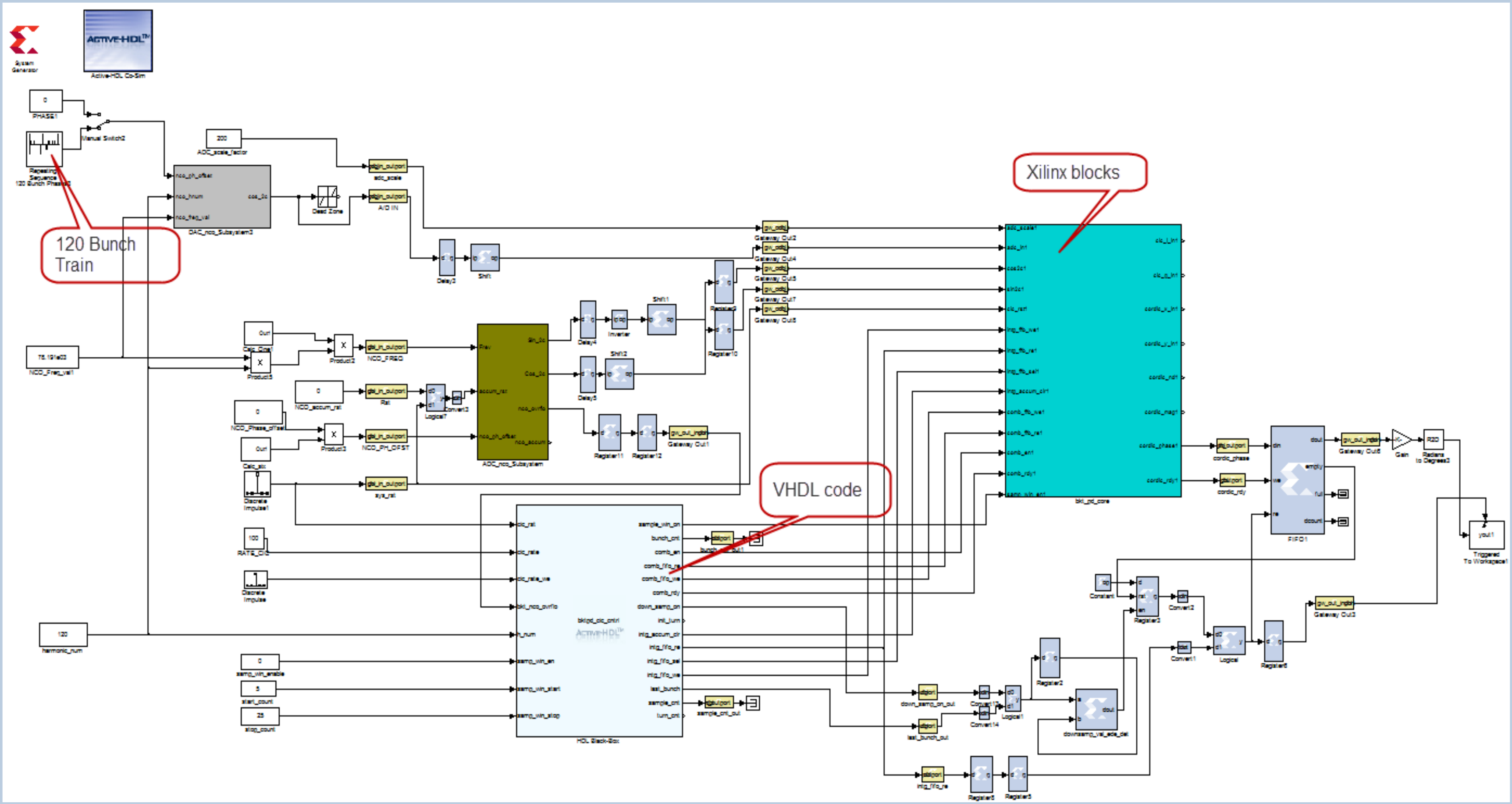
Bunch-by-Bunch Phase Detector IP



- Fast feedback bunch phase measurement

Ref: Kevin Smith - Overview of LLRF Developments at the BNL Collider-Accelerator Complex

Bunch-by-Bunch Phase Detector System



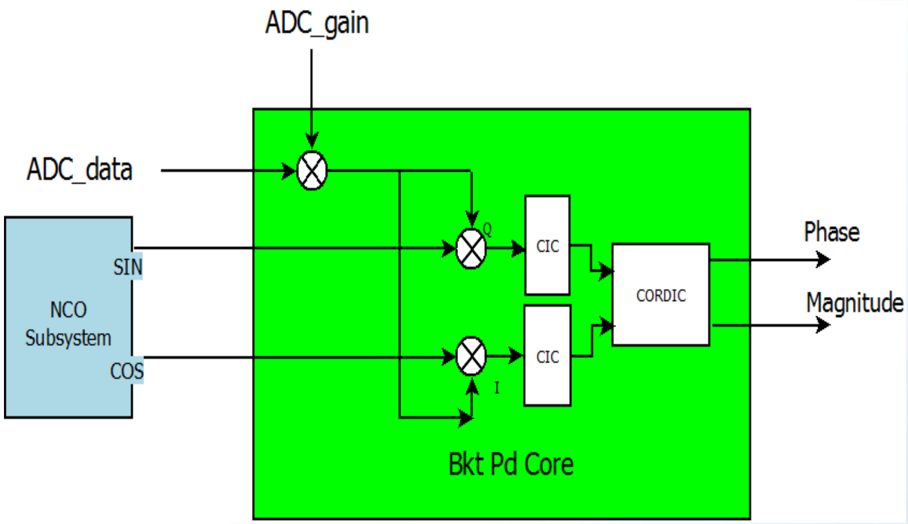
Building Blocks of DSP Datapath

- NCO Subsystem
  - LUT based
  - Tunable phase offset
  - Generate orthogonal reference signals for down conversion

- CIC (cascaded integrator comb) Filters
  - ideal for large rate changes and narrow band low pass filtering
  - implementation efficient - no multipliers, only addition and subtraction are needed.

- CORDIC

  many functional configurations:
  - vector rotation (polar to rectangular),
  - vector translation (rectangular to polar),
  - Sin and Cos
  - Sinh and Cosh
  - Atan and Atanh
  - Square Root



CIC Filter Specifications

- Narrow band low pass filter of consecutive bunches to determine phase and magnitude
- Bunch train at 9.386 MHz
- Support variable decimation rate R
- Processing clock 100 MHz
- A/D sampling rate 100 Msps
- FPGA resources limited
- Parameterizable Bit Growth
- Resolve inherent memory issues when time multiplexing bunches

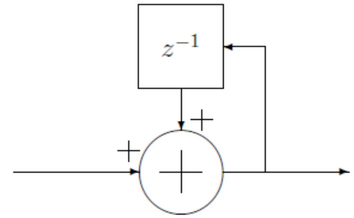


Figure 1: Basic Integrator

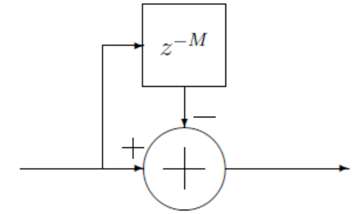


Figure 2: Basic Comb

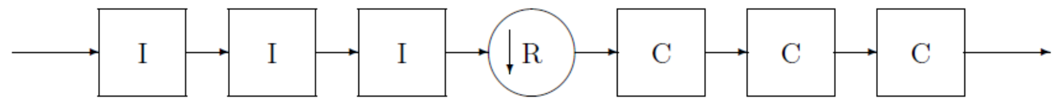
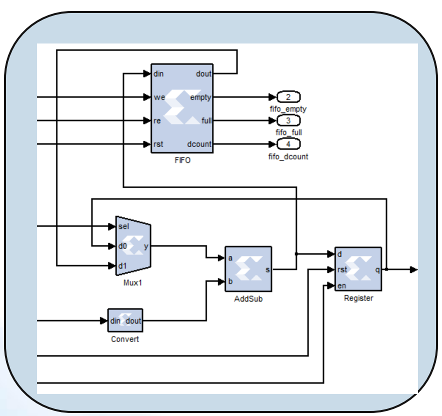


Figure 3: Three Stage Decimating CIC Filter

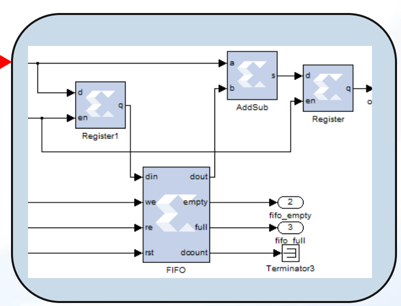
Implementation

- Single stage CIC sufficient for phase and magnitude detection
- Throughput meets requirement
- Decimation rate set before comb stage
- FIFOs to handle accumulator results from turn to turn
- CIC resource utilization (Virtex5)
  - Slice Reg = 200
  - BRAM = 1
  - DSP48E = 2

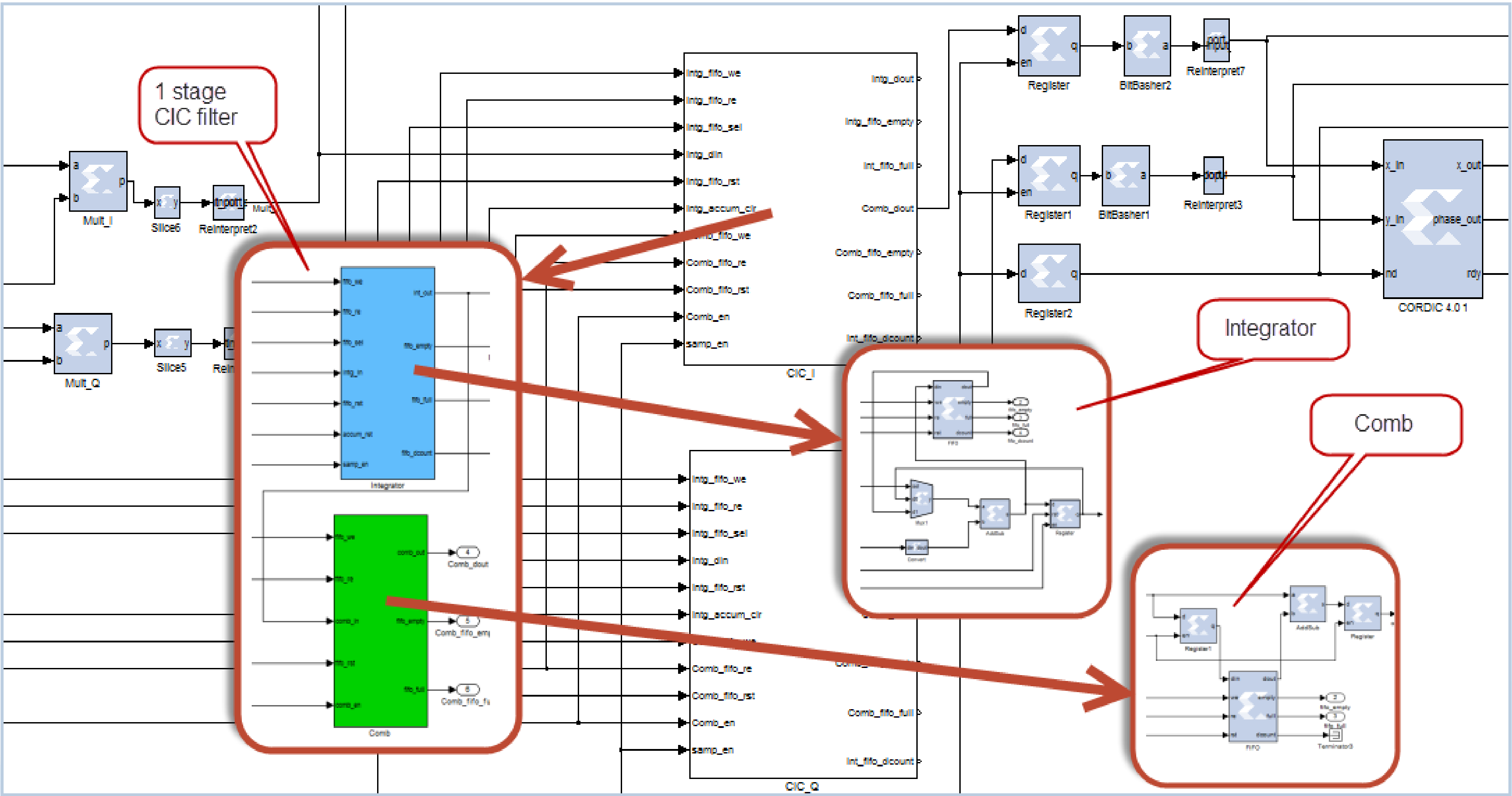


Integrator

Comb



Custom 1-Stage CIC filter



BROOKHAVEN NATIONAL LABORATORY

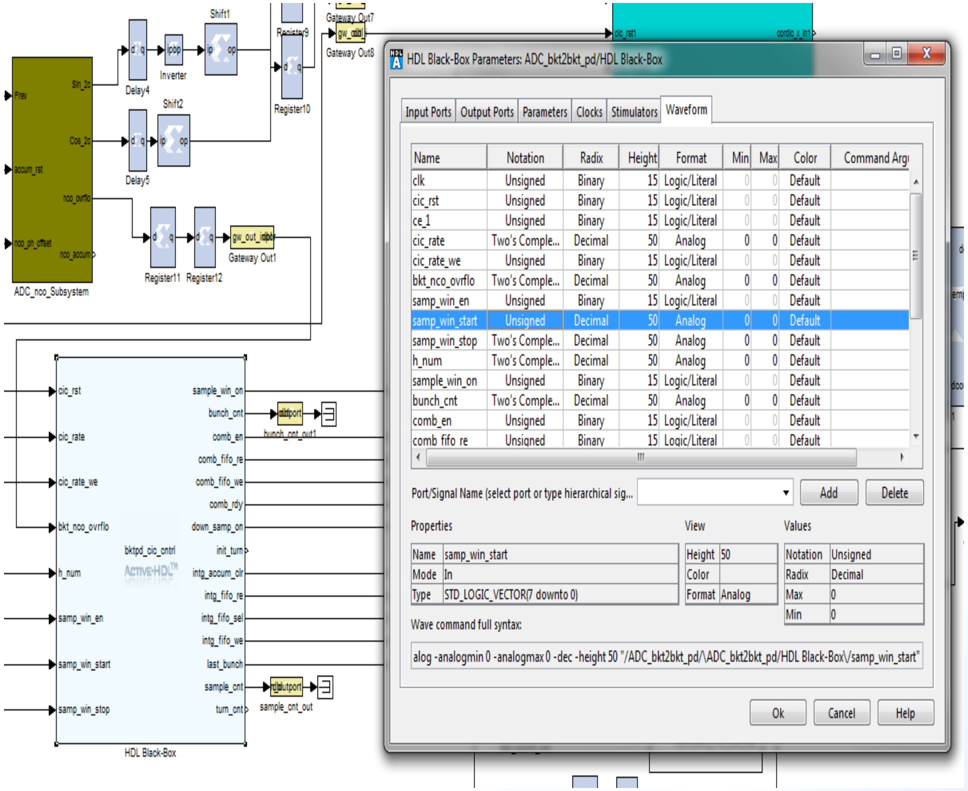
HDL Co-Simulation

- Legacy (or new) HDL code can be imported into Simulink “black box”

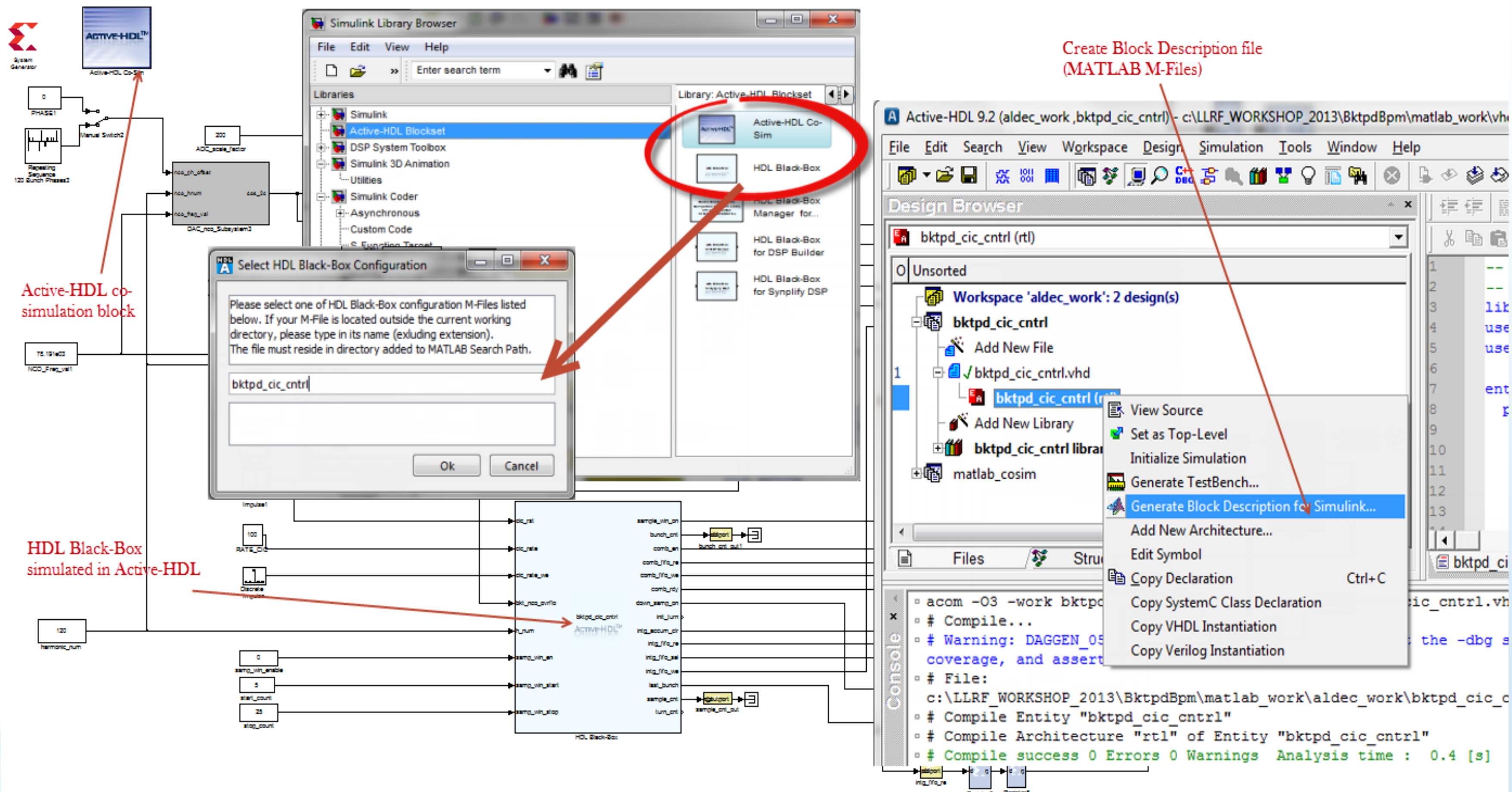
- HDL is co-simulated transparently using industry-standard simulators like ModelSim or Active-HDL

- Single HDL simulator for multiple black boxes

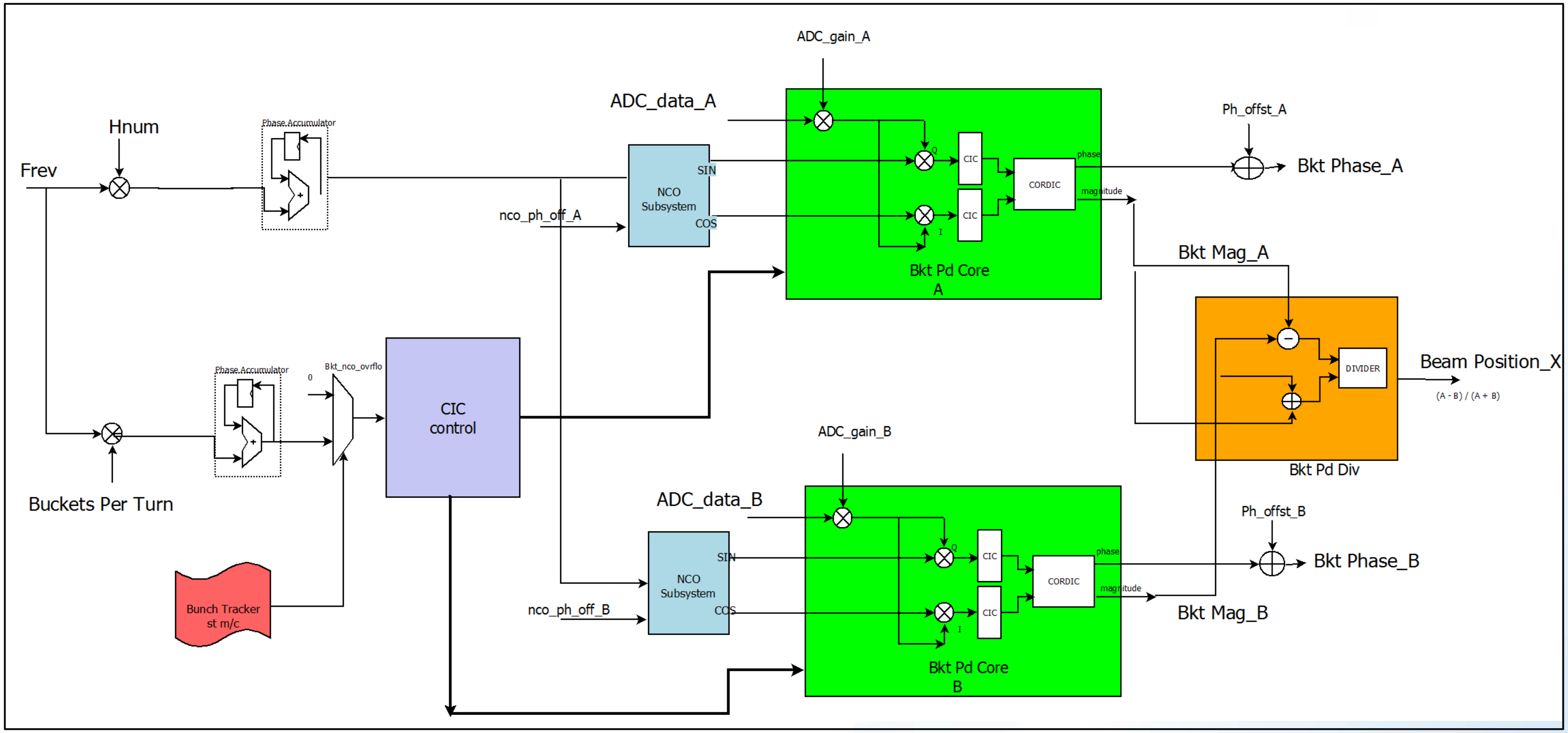
- The time scale in HDL Simulator matches that in Simulink



Configuring a HDL Black-Box to import RTL



Future Development
RF BPM Processor



Software Tools and Versions :

- MATLAB R2011a
  - Simulink
  - DSP System Toolbox
  - Signal Processing Design
  - Fixed-point Design
  - Fixed-point Blocks
- ISE Design Suite 13.2 System Edition
- Active-HDL 9.2

Demo Platform :

- SPARTAN-6 FPGA SP605 Evaluation Kit